Documentation for Remote Measurements Systems ADC-1

Version 0.07 16 May 1985 L. Shoer

I. Alphabetical list of routines in library

ACNTRL - Control BSR (115 VAC) devices from Remote Measurements ADC-1
ADCCOM - Primitive: Communicate with ADC-1 (send byte, receive byte)
ADCONV - Digitize an ADC-1 analog signal (from specified channel)
ADCPRM - Primitive: Perform a 12 bit analog-to-digital conversion
BSROUT - Primitive: Output a command to the BSR (115 VAC) controller
CONIN0 - Primitive: Input a byte from the console (8 bits)
CONIN1 - Primitive: Input a byte from the console, if available (8 bits)
CONOT0 - Primitive: Output a byte to the console (8 bits)
CNTRLI - Determine the status of the controlled inputs
CNTRLO - Set the status of the controlled outputs
OKUNIT - Determine if the specified ADC-1 unit is valid
TEMP - Determine the temperature in the ADC-1 onboard temperature sensor

II. Detailed discussion of routines in library

ACNTRL - Control AC (BSR) devices

SUBROUTINE ACNTRL(ADCUNT,DEVICE,FNCTN,VALID)
ADCUNT (BYTE) ADC-1 unit to control
DEVICE (BYTE) BSR device to control
Device is specified as units 1 to 32, where:
first house code, device 1 = 1
first house code, device 2 = 2
first house code, device 16 = 16
second house code, device 1 = 17
second house code, device 2 = 18
second house code, device 16 = 32
FNCTN (BYTE) BSR function to be performed
Clear = 0 (turns off all devices in last called house code)*
All = 1 (turns on all devices in last called house code)*
On = 2
Off = 3
Dim = 4
Bright = 5
VALID (LOGICAL) BSR command has been successfully transmitted

NOTE: There are peculirities in the way BSR devices work. The DIM and BRIGHT function both start with the device set on. If the device is initially off, the first call to BRIGHT turns the device on.
*The CLEAR and ALL functions are not fully checked out. They do not appear to have the meaning indicated above.

```
-------------------------------------------------------------------------

ADCCOM - Communicate with ADC-1 (send byte, receive byte)

SUBROUTINE ADCCOM(ADCUNT,OUTPUT,INPUT)
     ADCUNT (BYTE) ADC-1 unit to control
     OUTPUT (BYTE) datum to be transmitted to ADC-1
     INPUT (BYTE) datum returned by ADC-1

-------------------------------------------------------------------------

ADCONV - Digitize an analog signal

SUBROUTINE ADCONV(ADCUNT,CHANEL,INSTAM,DATUM,VALID)
     ADCUNT (BYTE) ADC-1 unit to control
     CHANEL (BYTE) Analog channel to digitize
          Channel 1 to 16 ( INSTAM = .FALSE. )
               Range: -0.4096 to +0.4096 volts
               Resolution: 0.0001 volts (100 microvolts)
                    voltage (volts)     DATUM
                         +0.4096        +4096
                          0.0               0
                         -0.4096        -4096
     INSTAM (LOGICAL) Instrumentation amplifier turned on=TRUE/off=FALSE
          Channel 1 to 8 only ( INSTAM = .TRUE. )
               Range: -0.04096 to +0.04096 volts
               Resolution: 0.00001 volts (10 microvolts)
                    voltage (volts)     DATUM
                         +0.04096       +4096
                          0.0               0
                         -0.04096       -4096
     DATUM (INTEGER) Result of analog-to-digital conversion
     VALID (LOGICAL) Valid digitized result is available
          May be FALSE if:
               Bad ADC-1 unit is requested
               Bad analog channel is requested
               Overvoltage has occurred

-------------------------------------------------------------------------

ADCPRM - Primitive: Perform a 12 bit analog-to-digital conversion

SUBROUTINE ADCPRM(ADCUNT,ADCCMD,DATUM,VALID)
     ADCUNT (BYTE) ADC-1 unit to control
     ADCCMD (BYTE) ADC command to be transmitted to ADC-1
     DATUM (INTEGER) Result of analog-to-digital conversion
     VALID (LOGICAL) Valid digitized result is available
          May be FALSE if:
               Bad ADC-1 unit is requested
               Bad analog channel is requested
               Overvoltage has occurred

-------------------------------------------------------------------------
```

```
BSROUT - Output a command to the AC (BSR) controller

SUBROUTINE BSROUT(ADCUNT,BSRCMD)
     ADCUNT (BYTE) ADC-1 unit to control
     BSRCMD (BYTE) BSR command to be transmitted to ADC-1

-------------------------------------------------------------------------

CNTRLI - Determine the status of the controlled inputs

SUBROUTINE CNTRLI(ADCUNT,LATCHI,RESET,VALID)
     ADCUNT (BYTE) ADC-1 unit to control
     LATCHI (LOGICAL(4)) Logical array containing status of cont. inputs
          .FALSE. = controlled input is off
          .TRUE. = controlled input is on
     RESET (LOGICAL) Reset latches to startup condition
          Latch 1 = normally closed (trigger occurs when opened)
          Latch 2 = normally closeed (trigger occurs when opened)
          Latch 3 = normally open (trigger occurs when closed)
          Latch 4 = normally open (trigger occurs when closed)
     VALID (LOGICAL) Command has been successfully transmitted

-------------------------------------------------------------------------

CNTRLO - Set the status of the controlled outputs

SUBROUTINE CNTRLO(ADCUNT,LATCHO,VALID)
     ADCUNT (BYTE) ADC-1 unit to control
     LATCHO (LOGICAL(6)) Logical array to set controlled outputs
          .FALSE. = set controlled output to off
          .TRUE. = set controlled output to on
     VALID (LOGICAL) Command has been successfully transmitted

-------------------------------------------------------------------------

CONIN0 - Input a single byte from the present console.  If no input is
         available, wait for input.

SUBROUTINE CONIN0(INPUT)
     INPUT (BYTE) Byte which is input (8 bits - no mask)

-------------------------------------------------------------------------

CONIN1 - Input a single byte from the present console.  If no input is
         available do not wait for input.

SUBROUTINE CONIN0(INPUT)
     INPUT (BYTE) Byte which is input (8 bits - no mask)
          If no input is available, INPUT = 00 .

-------------------------------------------------------------------------
```

CONOT0 - Output a single byte to the present console.

SUBROUTINE CONOT0(OUTPUT)
    OUTPUT (BYTE) Byte which is to be output (8 bits - no mask will be
                  used)

OKUNIT - Determine if the specified ADC-1 unit is valid

LOGICAL FUNCTION OKUNIT(ADCUNT)
    ADCUNT (BYTE) ADC-1 unit to control

    Function evaluates:
        TRUE - if ADC-1 unit is valid
        FALSE - if ADC-1 unit is not valid

TEMP - Determine the temperature in the ADC-1 onboard temperature sensor

SUBROUTINE TEMP(ADCUNT,DATUM,VALID)
    ADCUNT (BYTE) ADC-1 unit to control
    DATUM (INTEGER) Temperature result of analog-to-digital conversion
    VALID (LOGICAL) Valid digitized result is available
        May be FALSE if:
            Bad ADC-1 unit is requested
            Overvoltage has occurred

BSR (115 VAC) Control

A load must be applied to observe the following voltages.

OFF	0.1 VAC
ON	121.2 VAC

The increment associated with each DIM or BRIGHT command has not been accurately determined. Preliminary data is as follows:

volts	14.93	15.51	16.88	18.13	19.41	20.14	21.53
increment		0.58	1.37	1.25	1.28	0.73	1.39

volts	14.63	25.4	50.30	75.50	99.84
steps	0	11	35	61	89

```
      SUBROUTINE ACNTRL(ADCUNT,DEVICE,FNCTN,VALID)
C
C     VERSION 0.06     12 JULY 1984     L. SHOER
C     VERSION 1.06     30 DEC 84        B.BURLEIGH
C     VERSION 2.06     5 FEBRUARY 1985  L. SHOER
C
C     CONTROL BSR DEVICES FROM REMOTE MEASUREMENTS DEVICE
C
      EXTERNAL OKUNIT
C
      BYTE ADCUNT,BRIGHT,BSRCMD,BSRDEV,BSRFNC,CLEAR,CMDBAS
      BYTE DEVICE,DEVBAS,DEVCDS(16),FNCTN,HCOFST,HSECOD,INDEX
C
      LOGICAL OKUNIT,VALID
C
      DATA CLEAR,BRIGHT/
     1          0,     5/
C
      DATA CMDBAS,DEVBAS/
     1         190,   192/
C
      DATA DEVCDS/
C          1     2     3     4     5     6     7     8
     1     12,   28,    4,   20,    2,   18,   10,   26,
C
C          9    10    11    12    13    14    15    16
     2     14,   30,    6,   22,    0,   16,    8,   24/
C
C     INITIALIZE - SUCCESSFUL BSR CONTROL HAS NOT YET OCCURRED
            VALID=.FALSE.
C
C     CHECK IF VALID ADC-1 UNIT HAS BEEN REQUESTED
            IF (.NOT.OKUNIT(ADCUNT)) RETURN
C     CHECK IF VALID BSR DEVICE HAS BEEN REQUESTED
            IF ((DEVICE.LT.1).OR.(DEVICE.GT.32)) RETURN
C     CHECK IF VALID BSR FUNCTION HAS BEEN REQUESTED
            IF ((FNCTN.LT.CLEAR).OR.(FNCTN.GT.BRIGHT)) RETURN
C     EVALUATE "HOUSE CODE" AND OFFSET
            HSECOD=(DEVICE-1)/16+1
            HCOFST=(HSECOD-1)*32
C
C     EVALUATE BSR DEVICE
            INDEX=DEVICE-(HCOFST/2)
            BSRDEV=DEVBAS+DEVCDS(INDEX)+HCOFST
C     OUTPUT DEVICE TO REMOTE MEASUREMENTS DEVICE
            CALL BSROUT(ADCUNT,BSRDEV)
C
C     EVAULATE BSR FUNCTION
            BSRFNC=FNCTN+FNCTN+3
C     EVALUATE BSR COMMAND
            BSRCMD=CMDBAS+BSRFNC+HCOFST
C     OUTPUT COMMAND TO REMOTE MEASUREMENTS DEVICE
            CALL BSROUT(ADCUNT,BSRCMD)
C
C     INDICATE THAT A BSR COMMAND HAS BEEN TRANSMITTED
            VALID=.TRUE.
C
      RETURN
      END
```

```
      SUBROUTINE ADCCOM(ADCUNT,OUTPUT,INPUT)
C
C     VERSION 0.04     11 JULY 1984     L. SHOER
C     VERSION 0.05     10 OCT  1984     B.BURLEIGH
C     VERSION 1.05     5 FEBRUARY 1985  L. SHOER
C
C     COMMUNICATE WITH REMOTE MEASUREMENTS DEVICE
C
      BYTE ADCUNT,INPUT,NEWCON,OFFSET,OLDCON,OUTPUT
C
C     REMOTE MEASUREMENTS DEVICE ASSIGNMENTS
C            ADC-1 UNIT 1 = CONSOLE 11
C            ADC-1 UNIT 2 = CONSOLE 12
C
            DATA OFFSET/10/
C
C     OBTAIN PRESENT CONSOLE ASSIGNMENT
            CALL GETCON(OLDCON)
C
C     CALCULATE NEW CONSOLE ASSIGNMENT
            NEWCON=ADCUNT+OFFSET
C
C     SET NEW CONSOLE ASSIGNMENT
            CALL SETCON(NEWCON)
C
C     OUTPUT DATA
            CALL CONOT0(OUTPUT)
C
C     INPUT DATA
            CALL CONIN0(INPUT)
C
C     RESTORE ORIGINAL CONSOLE ASSIGNMENT
            CALL SETCON(OLDCON)
C
      RETURN
      END
```

```
      SUBROUTINE ADCONV(ADCUNT,CHANEL,INSTAM,DATUM,VALID)
C
C     VERSION 0.02     4 JULY 1984     L. SHOER
C     VERSION 1.02     30 DEC 84       B.BURLEIGH
C
C     12 BIT ANALOG-TO-DIGITAL CONVERSION
C     REMOTE MEASUREMENTS DEVICE
C
      EXTERNAL OKUNIT
C
      BYTE ADCCMD,ADCUNT,BASE,CHANEL
C
      INTEGER DATUM
C
      LOGICAL VALID,INSTAM,OKUNIT
C
C     INITIALIZE - DATA CONVERSION IS NOT YET VALID
            VALID=.FALSE.
C
C     CHECK IF VALID ADC-1 UNIT HAS BEEN REQUESTED
            IF (.NOT.OKUNIT(ADCUNT)) RETURN
C
C     CHECK IF INVALID CHANNEL IS REQUESTED
C           CHANNELS 1 TO 16 ARE VALID
                  IF ((CHANEL.LT.1).OR.(CHANEL.GT.16)) RETURN
C           ONLY CHANNELS 1 TO 8 ARE VALID IF USING INSTR. AMP.
                  IF ((CHANEL.GT.8).AND.INSTAM) RETURN
C
C     DERIVE COMMAND FOR REMOTE MEASUREMENTS DEVICE
            BASE=0
            IF (INSTAM) BASE=16
            ADCCMD=BASE+CHANEL-1
C
C     PERFORM ANALOG-TO-DIGITAL CONVERSION
            CALL ADCPRM(ADCUNT,ADCCMD,DATUM,VALID)
C
      RETURN
      END
```

```
      SUBROUTINE ADCPRM(ADCUNT,ADCCMD,DATUM,VALID)
C
C     VERSION 0.01     12 JULY 1984     L. SHOER
C     VERSION 1.01     30 DEC 84        B.BURLEIGH
C     VERSION 2.02     8 FEBRUARY 1985  L. SHOER
C
C     12 BIT ANALOG-TO-DIGITAL CONVERSION
C
      BYTE ADCCMD,ADCUNT,MASK,HIBYTE,HIST,INPROG,LOBYTE
      BYTE OVRVLT,POSSGN,RSPNCE,SNHIST,SNLOBY
C
      INTEGER DATUM,HIINT,LOINT,SIGN
C
      LOGICAL VALID
C
C
C     FOR THE FOLLOWING ASSIGNMENTS, SEE MANUAL: P. C-1
C
            DATA MASK,       INPROG,     OVRVLT,     POSSGN/
C               00001111    10000000    00100000    00010000
     1          15,         128,        32,         16/
C
            DATA SNHIST,SNLOBY/
     1             161,   145/
C
C
C     INITIALIZE - DATA CONVERSION IS NOT YET VALID
            VALID=.FALSE.
C
C     START CONVERSION
            CALL ADCCOM(ADCUNT,ADCCMD,RSPNCE)
C
C     REQUEST STATUS (AND HI BYTE) FROM REMOTE MEASUREMENTS DEVICE
100         CALL ADCCOM(ADCUNT,SNHIST,HIST)
C
C     IF CONVERSION IS NOT COMPLETE, WAIT
            IF(HIST.AND.INPROG)GO TO 100
C
C     IF OVERVOLTAGE CONDITION, RETURN WITH VALID STILL SET FALSE
            IF (HIST.AND.OVRVLT) RETURN
C
C     DETERMINE SIGN OF RESULT
            SIGN=-1
            IF (HIST.AND.POSSGN) SIGN=+1
C
C     EVALUATE HI BYTE
            HIINT=HIST.AND.MASK
C
C     REQUEST LO BYTE FROM REMOTE MEASUREMENTS DEVICE
            CALL ADCCOM(ADCUNT,SNLOBY,LOBYTE)
C
C     EVALUATE LO BYTE
C
C
C           IN MICROSOFT FORTRAN 1 BYTE INTGER IS STORED AS TWO'S
C           COMPLEMENT.  RANGE = -128 TO +127
C           2 BYTE INTEGER IS ALSO STORED AS TWO'S COMPLEMENT.
C
C           BIT PATTERN:     00000000  TO  01111111
C               EQUALS:             0  TO      +127
C           BIT PATTERN:     10000000  TO  11111111
C               EQUALS:          -128  TO        -1
C
            LOINT=LOBYTE
            IF (LOINT.LT.0) LOINT=LOINT+256
C
C     COMBINE HIBYTE, LOBYTE, AND SIGN
            DATUM=SIGN*(HIINT*256+LOINT)
C
C     INDICATE THAT A SUCCESFUL CONVERSION HAS TAKEN PLACE
            VALID=.TRUE.
C
      RETURN
      END
```

```
      SUBROUTINE BSROUT(ADCUNT,BSRCMD)
C
C     VERSION 0.02     5 JULY 1984      L. SHOER
C     VERSION 1.02     30 DEC 84        B.BURLEIGH
C     VERSION 2.02     5 FEBRUARY 1985  L. SHOER
C
C     OUTPUT BYTE TO REMOTE MEASUREMENTS DEVICE
C
      BYTE ADCUNT,BSRCMD,BUSY,RQSTAT,RSPNCE,STATUS
C
C     FOR THE FOLLOWING ASSIGNMENTS, SEE MANUAL: P. 24, B-17, C-1
            DATA    BUSY,           RQSTAT/
C                   00010000        10000001
     1              16,               129/
C
C
C     OUTPUT COMMAND TO REMOTE MEASUREMENTS DEVICE
            CALL ADCCOM(ADCUNT,BSRCMD,RSPNCE)
C
C     WAIT FOR TRANSMISSION OF COMMAND TO BE COMPLETED
C           1 TRANSMISSION = 0.4 SEC
C           TRANSMISSION IS REPEATED AT LEAST 3 TIMES FOR ACCURACY
C           MINIMUM TRANSMISSION TIME IS (3 * 0.4 SEC) = 1.2 SEC
C           NAP(1) IS A 0.02 SEC DELAY
C           WAIT 1.2 SEC
                  CALL NAP(60)
C
C     REQUEST STATUS FROM REMOTE MEASUREMENTS DEVICE
100         CALL ADCCOM(ADCUNT,RQSTAT,STATUS)
C
C     WAIT FOR COMMAND TO BE CARRIED OUT
            IF (STATUS.AND.BUSY) GO TO 100
C
      RETURN
      END
```

```
      SUBROUTINE CNTRLI(ADCUNT,LATCHI,RESET,VALID)
C
C     VERSION 0.01     5 JULY 1984     L. SHOER
C     VERSION 1.01     30 DEC 84       B.BURLEIGH
C
C     READ CONTROLLED INPUT LATCHES ON REMOTE MEASUREMENTS DEVICE
C           INPUTS ARE IDENTIFIED BY THEIR INDIVIDUAL BIT
C              4         3         2         1
C           00001000  00000100  00000010  00000001
C
      EXTERNAL OKUNIT
C
      BYTE ADCUNT,NMBR,PNTR,RQSTAT,RSTLAT,STATUS
      LOGICAL LATCHI(4),OKUNIT,RESET,VALID
C
      DATA RSTLAT,RQSTAT/
     1        128,   129/
C
C     INITIALIZE
C           INPUT LATCHES ARE NOT YET VALID
                  VALID=.FALSE.
C           (RE)SET ALL LATCH VARIABLES TO FALSE
                  DO 100 NMBR=1,4
                        LATCHI(NMBR)=.FALSE.
100               CONTINUE
C
C     CHECK IF VALID ADC-1 UNIT HAS BEEN REQUESTED
          IF (.NOT.OKUNIT(ADCUNT)) RETURN
C
C     DETERMINE IF LATCHES TO BE RESET
          IF (.NOT.RESET) GO TO 150
C         RESET LATCHES
                  CALL ADCCOM(ADCUNT,RSTLAT,STATUS)
C     ******
C     CALL PRBYTE(RSTLAT)
C     ******
                  GO TO 200
C         LATCHES NOT TO BE RESET, JUST REQUEST STATUS
150               CALL ADCCOM(ADCUNT,RQSTAT,STATUS)
C     ******
C     CALL PRBYTE(RQSTAT)
C     ******
C
C     DETERMINE STATUS OF INDIVIDUAL INPUTS
200         DO 300 NMBR=1,4
                  PNTR=NMBR-1
                  IF (STATUS.AND.2**PNTR) LATCHI(NMBR)=.TRUE.
300         CONTINUE
C
C     INDICATE THAT INPUTS HAVE BEEN EXAMINED
          VALID=.TRUE.
C
      RETURN
      END
```

```
      SUBROUTINE CNTRLO(ADCUNT,LATCHO,VALID)
C
C     VERSION 0.03     5 JULY 1984     L. SHOER
C     VERSION 1.03     30 DEC 84       B.BURLEIGH
C
C     CONTROL OUTPUT LATCHES ON REMOTE MEASUREMENTS DEVICE
C           OUTPUTS ARE IDENTIFIED BY THEIR INDIVIDUAL BIT
C             BASE
C           01000000
C              6         5         4         3         2         1
C           00100000  00010000  00001000  00000100  00000010  00000001
C
      EXTERNAL OKUNIT
C
      BYTE ADCUNT,ADCCMD,NMBR,PNTR,RSPNCE
      LOGICAL LATCHO(6),OKUNIT,VALID
C
C     INITIALIZE
C           OUTPUT LATCHES NOT YET CONTROLLED
                 VALID=.FALSE.
C           BASE ADDRESS FOR CONTROL OF OUTPUT LATCHES
                 ADCCMD=64
C
C     CHECK IF VALID ADC-1 UNIT HAS BEEN REQUESTED
           IF (.NOT.OKUNIT(ADCUNT)) RETURN
C
C     EVALUATE ADC-1 COMMAND
           DO 100 NMBR=1,6
                 PNTR=NMBR-1
                 IF (LATCHO(NMBR)) ADCCMD=ADCCMD+2**PNTR
100        CONTINUE
C
C     ****************************************************************
C          WRITE(1,900)LATCHO,ADCCMD
C900       FORMAT(' LATCHO:',6(L1,4X),' ADCCMD:',I5)
C     ****************************************************************
C
C     OUTPUT COMMAND TO REMOTE MEASUREMENTS DEVICE
           CALL ADCCOM(ADCUNT,ADCCMD,RSPNCE)
C
C     INDICATE THAT A COMMAND HAS BEEN TRANSMITTED
           VALID=.TRUE.
C
      RETURN
      END
```

```
                        ;CONIN0
                        ;Console input
                        ;version 0.00    9 October 1984    L. Shoer
                        ;
                        ;Input a single byte from cosole
                        ;If no input is available, wait for input
                        ;
                        ;This routine can be accessed through FORTRAN as a subroutine:
                        ;
                        ;       CALL CONIN0(BYTE)
                        ;
                        ;       BYTE is a byte
                        ;
                                .8080                   ;8080 OPCODES
0000'                           CSEG                    ;CODE RELATIVE MODE
                        ;
                                PUBLIC  CONIN0          ;FORTRAN SUBROUTINE CALL TO THIS NAME
                        ;
0005                    BDOS    EQU     05H     ;BDOS ENTRY POINT
0003                    RAWIN   EQU     03H     ;MPM FUNCTION 3 - RAW CONSOLE INPUT
                        ;
                        ;SAVE REGISTERS WHICH WILL BE ALTERED
0000' F5                CONIN0: PUSH    PSW
0001' C5                        PUSH    B
0002' D5                        PUSH    D
0003' E5                        PUSH    H
                        ;
                        ;GET A BYTE FROM THE CONSOLE
0004' 0E 03                     MVI     C,RAWIN         ;RAW CONSOLE INPUT FUNCTION
0006' CD 0005                   CALL    BDOS            ;CONSOLE INPUT RETURNED IN REG. A
                        ;
                        ;SAVE THE ENTERED KEY ~(OR 00 IF NO KEY PRESSED)~
0009' E1                        POP     H               ;LOCATION TO SAVE BYTE (FROM FORTRAN)
000A' 77                        MOV     M,A             ;SAVE BYTE
                        ;
                        ;RESTORE REGISTERS WHICH HAVE BEEN ALTERED
000B' D1                        POP     D
000C' C1                        POP     B
000D' F1                        POP     PSW
                        ;
000E' C9                        RET
                        ;
                                END
```

Macros:

Symbols:

0005	BDOS	0000I'	CONIN0	0003	RAWIN

No Fatal error(s)

```
                        ;CONIN1
                        ;Console input without wait
                        ;version 0.04     7 October 1984     L. Shoer
                        ;
                        ;Single key pressed returned to calling routine
                        ;If no key is pressed, null (00) is returned
                        ;
                        ;This routine can be accessed through FORTRAN as a subroutine:
                        ;
                        ;       CALL CONIN1(KEY)
                        ;
                        ;       KEY is a byte
                        ;
                                .8080                   ;8080 OPCODES
0000'                           CSEG                    ;CODE RELATIVE MODE
                        ;
                                PUBLIC  CONIN1          ;FORTRAN SUBROUTINE CALL TO THIS NAME
                        ;
0005                    BDOS    EQU     05H     ;BDOS ENTRY POINT
0006                    DIRCON  EQU     06H     ;CPM FUNCTION 6 - DIRECT CONSOLE I/O
00FF                    INPFNC  EQU     0FFH    ;       SUBSET - INPUT FUNCTION
                        ;
                        ;SAVE REGISTERS WHICH WILL BE ALTERED
0000'  F5               CONIN1: PUSH    PSW
0001'  C5                       PUSH    B
0002'  D5                       PUSH    D
0003'  E5                       PUSH    H
                        ;
                        ;GET A KEY FROM THE CONSOLE
0004'  0E 06                    MVI     C,DIRCON        ;DIRECT CONSOLE I/O FUNCTION
0006'  1E FF                    MVI     E,INPFNC        ;INPUT A CHARACTER FUNCTION
0008'  CD 0005                  CALL    BDOS            ;KEYBOARD INPUT RETURNED IN REG. A
                        ;
                        ;SAVE THE ENTERED KEY (OR 00 IF NO KEY PRESSED)
000B'  E1                       POP     H               ;LOCATION TO SAVE KEY (FROM FORTRAN)
000C'  77                       MOV     M,A             ;SAVE KEY
                        ;
                        ;RESTORE REGISTERS WHICH HAVE BEEN ALTERED
000D'  D1                       POP     D
000E'  C1                       POP     B
000F'  F1                       POP     PSW
                        ;
0010'  C9                       RET
                        ;
                                END
```

Macros:

Symbols:

0005	BDOS	0000I'	CONINI	0006	DIRCON
00FF	INPFNC				

No Fatal error(s)

not presently in library


```
                        ;CONIN2
                        ;Console input using BDOS Function 6
                        ;version 0.00     24 October 1984     L. Shoer
                        ;
                        ;Single byte returned to calling routine
                        ;Routine waits for byte to be available
                        ;
                        ;This routine can be accessed through FORTRAN as a subroutine
                        ;
                        ;       CALL CONIN2(BYTE)
                        ;
                        ;       BYTE is a byte
                        ;
                                .8080                   ;8080 OPCODES
0000'                           CSEG                    ;CODE RELATIVE MODE
                        ;
                                PUBLIC  CONIN2          ;FORTRAN SUBROUTINE CALL TO THIS NAME
                        ;
0005                            BDOS    EQU     05H     ;BDOS ENTRY POINT
0006                            DIRCON  EQU     06H     ;CPM FUNCTION 6 - DIRECT CONSOLE I/O
00FD                            INPFNC  EQU     0FDH    ;       SUBSET - INPUT BYTE (WAIT)
                        ;
                        ;       SAVE REGISTERS WHICH WILL BE ALTERED
0000'  F5               CONIN2: PUSH    PSW
0001'  C5                       PUSH    B
0002'  D5                       PUSH    D
0003'  E5                       PUSH    H
                        ;
                        ;GET A BYTE FROM THE CONSOLE
0004'  0E 06                    MVI     C,DIRCON        ;DIRECT CONSOLE I/O FUNCTION
0006'  1E FD                    MVI     E,INPFNC        ;WAIT FOR INPUT BYTE
0008'  CD 0005                  CALL    BDOS            ;BYTE RETURNED IN REG. A
                        ;
                        ;SAVE THE BYTE
000B'  E1                       POP     H               ;LOCATION TO SAVE BYTE (FROM FORTRAN)
000C'  77                       MOV     M,A             ;SAVE BYTE
                        ;
                        ;RESTORE REGISTERS WHICH HAVE BEEN ALTERED
000D'  D1                       POP     D
000E'  C1                       POP     B
000F'  F1                       POP     PSW
                        ;
0010'  C9                       RET
                        ;
                                END
```

Macros:

Symbols:

0005	BDOS	0000I'	CONIN2	0006	DIRCON
00FD	INPFNC				

No Fatal error(s)

```
                        ;CONOT0
                        ;Console output
                        ;version 0.00     9 October 1984     L. Shoer
                        ;
                        ;Output a single byte to cosole
                        ;
                        ;This routine can be accessed through FORTRAN as a subroutine:
                        ;
                        ;       CALL CONOT0(BYTE)
                        ;
                        ;       BYTE is a byte
                        ;
                                .8080                   ;8080 OPCODES
0000'                           CSEG                    ;CODE RELATIVE MODE
                        ;
                                PUBLIC  CONOT0          ;FORTRAN SUBROUTINE CALL TO THIS NAME
                        ;
0005                            BDOS    EQU     05H     ;BDOS ENTRY POINT
0004                            RAWOUT  EQU     04H     ;MPM FUNCTION 4 - RAW CONSOLE OUTPUT
                        ;
                        ;SAVE REGISTERS WHICH WILL BE ALTERED
0000'  F5               CONOT0: PUSH    PSW
0001'  C5                       PUSH    B
0002'  D5                       PUSH    D
0003'  E5                       PUSH    H
                        ;
                        ;OUTPUT A BYTE TO THE CONSOLE
0004'  0E 04                    MVI     C,RAWOUT        ;RAW CONSOLE OUTPUT FUNCTION
0006'  5E                       MOV     E,M             ;BYTE PLACED IN REG. E (FROM FORTRAN)
0007'  CD 0005                  CALL    BDOS            ;OUTPUT THE BYTE
                        ;
                        ;RESTORE REGISTERS WHICH HAVE BEEN ALTERED
000A'  E1                       POP     H
000B'  D1                       POP     D
000C'  C1                       POP     B
000D'  F1                       POP     PSW
                        ;
000E'  C9                       RET
                        ;
                                END
```

Macros:

Symbols:

0005	BDOS	0000I'	CONOT0	0004	RAWOUT

No Fatal error(s)

```
      LOGICAL FUNCTION OKUNIT(ADCUNT)
C
C     VERSION 0.00     4 JULY 1984     L. SHOER
C     VERSION 1.00     30 DEC 84       B.BURLEIGH
C
C     DETERMINE IF ADC-1 UNIT NUMBER IS VALID
C            THERE ARE PRESENTLY TWO REMOTE MEASUREMENTS DEVICES (ADC-1)
C            THESE UNITS ARE DESIGNATED NUMBERS 1 AND 2
C
      BYTE ADCUNT
      LOGICAL OKUNIT
C
C     INITIALIZE - ADC-1 UNIT IS NOT YET VALID
            OKUNIT=.FALSE.
C
      IF ((ADCUNT.GE.1).AND.(ADCUNT.LE.2)) OKUNIT=.TRUE.
C
      RETURN
      END
```

```
      SUBROUTINE TEMP(ADCUNT,DATUM,VALID)
C
C     VERSION 0.01     4 JULY 1984     L. SHOER
C     VERSION 1.01     30 DEC 84       B.BURLEIGH
C
C     OBTAIN TEMPERATURE FROM ONBOARD SENSOR OF REMOTE MEASUREMENTS DEVICE
C
      EXTERNAL OKUNIT
C
      BYTE ADCUNT,RQTEMP
      INTEGER DATUM
      LOGICAL OKUNIT,VALID
C
      DATA RQTEMP/24/
C
C     INITIALIZE - TEMPERATURE NOT YET VALID
            VALID=.FALSE.
C
C     CHECK IF VALID ADC-1 UNIT HAS BEEN REQUESTED
            IF (.NOT.OKUNIT(ADCUNT)) RETURN
C
C     OBTAIN TEMPERATURE
            CALL ADCPRM(ADCUNT,RQTEMP,DATUM,VALID)
C
      RETURN
      END
```

```
A:PIP $1=$2ADCLIB.DOC
A:PIP $1=$2ADCPIP.SUB
A:PIP $1=$2ADCF80.SUB
A:PIP $1=$2ACNTRL.FOR
A:PIP $1=$2ADCCOM.FOR
A:PIP $1=$2ADCONV.FOR
A:PIP $1=$2ADCPRM.FOR
A:PIP $1=$2BSROUT.FOR
A:PIP $1=$2CNTRLI.FOR
A:PIP $1=$2CNTRLO.FOR
A:PIP $1=$2OKUNIT.FOR
A:PIP $1=$2TEMP.FOR
```

```
A:F80 =$1ACNTRL
A:F80 =$1ADCCOM
A:F80 =$1ADCONV
A:F80 =$1ADCPRM
A:F80 =$1BSROUT
A:F80 =$1CNTRLI
A:F80 =$1CNTRLO
A:F80 =$1OKUNIT
A:F80 =$1TEMP
```

```
LIB80 $1ADCLB1=$1CNTRLO,$1CNTRLI,$1ACNTRL,$1ADCONV,$1TEMP/E
LIB80 $1ADCLB2=$1OKUNIT,$1BSROUT,$1ADCPRM,$1ADCCOM/E
LIB80 $1ADCLB3=$1CONIN0,$1CONOT0,$1CONIN1/E
LIB80 $1ADCLIB=$1ADCLB1,$1ADCLB2,$1ADCLB3/E
ERA $1ADCLB?.REL
```

```
.FI ADCLIB.DOC
.PA
.FI ACNTRL.FOR
.PA
.FI ADCCOM.FOR
.PA
.FI ADCONV.FOR
.PA
.FI ADCPRM.FOR
.PA
.FI BSROUT.FOR
.PA
.FI CNTRLI.FOR
.PA
.FI CNTRLO.FOR
.PA
.FI OKUNIT.FOR
.PA
.FI TEMP.FOR
.PA
```

```
      PROGRAM TSTADC
C
C     VERSION 0.01     11 JULY 1984     L. SHOER
C
C     TEST ANALOG-TO-DIGITAL CONVERSION ON REMOTE MEASUREMENTS DEVICES
C
      BYTE ADCUNT,BELL,CHANEL
      INTEGER DATUM
      LOGICAL INSTAM,VALID
C
      DATA BELL/07/
C
C     CLEAR SCREEN
50          CALL CLRSCR
C     OBTAIN ADC-1 UNIT PARAMETERS
            WRITE(1,150)
150         FORMAT(' ADC-1 unit for analog-to-digital conversion? ')
            READ(1,200)ADCUNT
200         FORMAT(I3)
            WRITE(1,210)
210         FORMAT(' ADC-1 channel to monitor? (1-16 or 1-8) ')
            READ(1,200)CHANEL
            WRITE(1,220)
220         FORMAT(' Turn on Instrumentation Amplifier? (T or F) ')
            READ(1,230)INSTAM
230         FORMAT(1L1)
C
C     PERFORM FUNCTION
255         CALL ADCONV(ADCUNT,CHANEL,INSTAM,DATUM,VALID)
            WRITE(1,260)DATUM
260         FORMAT('+Datum is',I6)
            IF (VALID) GO TO 255
C     OPERATION NOT SUCCESSFUL
            WRITE(1,300)BELL
300         FORMAT('+',1A1,//)
            PAUSE
            GO TO 50
C
      END
```

```
      PROGRAM TSTBSR
C
C     VERSION 0.06     5 FEBRUARY 1985     L. SHOER
C
C     TEST BSR DEVICES - 115 VAC CONTROL
C
      BYTE ADCUNT,BELL,DEVICE,FNCTN,TRMINL
      LOGICAL VALID
C
C     TERMINAL IN USE SPECIFIED BY TRMINL
            COMMON /TBYTE/TRMINL
C
      DATA BELL/07/
C
C     TERMINAL IN USE IS A VISUAL 500
            DATA TRMINL/05/
C
C     CLEAR SCREEN
50          CALL CLRSCR
C
C     DISPLAY MENU
            WRITE(1,100)
100         FORMAT(
     1            ' Control BSR modules',//
     2            '     ADC-1 units  = 1 or 2',//
     3            '     Devices 1 through 32',//
     4            '     Clear   = 0',/
     5            '     All     = 1',/
     6            '     On      = 2',/
     7            '     Off     = 3',/
     8            '     Dim     = 4',/
     9            '     Bright  = 5',//
     0            '     Restart = 6',//
     1            '     Quit    = 7',/)
C     OBTAIN ADC-1 UNIT TO CONTROL
140         WRITE(1,150)
150         FORMAT(' ADC-1 unit to control? (1 or 2)  ')
            READ(1,200)ADCUNT
200         FORMAT(I3)
            IF ((ADCUNT.LT.1).OR.(ADCUNT.GT.2)) GO TO 140
C
C     OBTAIN DEVICE TO CONTROL
240         WRITE(1,250)
250         FORMAT(' Device to control? (1 through 32)  ')
            READ(1,200)DEVICE
            IF ((DEVICE.LT.1).OR.(DEVICE.GT.32)) GO TO 240
C
C     GET FUNCTION
260               WRITE(1,275)
275               FORMAT(' BSR function? (0 through 7)  ',/)
C           GET A SINGLE BYTE FROM THE CONSOLE
C           (RETURN NOT NECESSARY)
280               CALL CONIN(FNCTN)
C           TURN BYTE INTO A FORTRAN INTEGER
                  FNCTN=FNCTN-48
C           IF FUNCTION IS NOT VALID, GET ANOTHER BYTE
                  IF ((FNCTN.LT.1).OR.(FNCTN.GT.7)) GO TO 280
C           IF FUNCTION IS QUIT, END PROGRAM
                  IF (FNCTN.EQ.7) GO TO 1000
C           IF FUNCTION IS RESTART, BACK TO BEGINNING
                  IF (FNCTN.EQ.6) GO TO 50
C
C     PERFORM FUNCTION
            CALL ACNTRL(ADCUNT,DEVICE,FNCTN,VALID)
```

```
C       OPERATION NOT SUCCESSFUL
              WRITE(1,300)BELL
300           FORMAT('+',1A1,'ERROR: BSR control not possible',//)
              CALL PAUSE
              GO TO 50
C
1000    END
```

```
BC(TSTBSR);
IN(TSTBSR);
SR(ADCLIB,A:PLIB1,A:AUXLIB,A:XFORLIB);
FI;
EN;
```

```
      PROGRAM TSTCI
C
C     VERSION 0.04     17 MAY     L. SHOER
C
C     TEST CONTROLLED INPUTS ON REMOTE MEASUREMENTS DEVICE
C
      BYTE ADCUNT,BELL
      LOGICAL LATCHI(4),RESET,VALID
C
      DATA BELL/07/
C
C     CLEAR SCREEN
            CALL CLRSCR
C
C     OBTAIN ADC-1 UNIT PARAMETERS
            WRITE(1,100)
100         FORMAT(' ADC-1 unit to report? ')
            READ(1,200)ADCUNT
200         FORMAT(I3)
C
C     RESET LATCHES?
            WRITE(1,300)
300         FORMAT(' Reset all latches? (T or F) ')
            READ(1,400)RESET
400         FORMAT(1L1)
            IF (.NOT.RESET) GO TO 500
C
C     PERFORM RESET
            CALL CNTRLI(ADCUNT,LATCHI,RESET,VALID)
C
C     REPORT STATUS OF LATCHES
500         RESET=.FALSE.
600         CALL CNTRLI(ADCUNT,LATCHI,RESET,VALID)
            IF (.NOT.VALID) GO TO 2000
            WRITE(1,700)LATCHI
700         FORMAT('+Latch status (1 through 4): ',4(L1,4X),/)
            CALL SLEEP(1)
            GO TO 600
C     OPERATION NOT SUCCESSFUL
2000        WRITE(1,2100)BELL
2100        FORMAT('+',1A1,//' Controlled output error',//)
C
5000  END
```

```
      PROGRAM TSTCO
C
C     VERSION 0.01     11 JULY 1984     L. SHOER
C
C     TEST CONTROLLED OUTPUTS ON REMOTE MEASUREMENTS DEVICE
C
      BYTE ADCUNT,BELL,LATCH
      LOGICAL LATCHO(6),VALID
C
      DATA BELL/07/
C
C     CLEAR SCREEN
50          CALL CLRSCR
C     SET ALL LATCHES OFF
            DO 70 I=1,6
70          LATCHO(I)=.FALSE.
C     OBTAIN ADC-1 UNIT PARAMETERS
100         WRITE(1,150)
150         FORMAT(' ADC-1 unit to control? ')
            READ(1,200)ADCUNT
200         FORMAT(I3)
205         WRITE(1,210)
210         FORMAT(' Latch to set? ')
            READ(1,200)LATCH
            WRITE(1,220)LATCH
220         FORMAT(' Set latch',I3,/
     1             ' On = True (T) or Off = False (F)? ')
            READ(1,230)LATCHO(LATCH)
230         FORMAT(1L1)
C
C     PERFORM FUNCTION
255         CALL CNTRLO(ADCUNT,LATCHO,VALID)
            IF (VALID) GO TO 205
C     OPERATION NOT SUCCESSFUL
            WRITE(1,300)BELL
300         FORMAT('+',1A1,//)
            PAUSE
            GO TO 50
C
      END
```

```
      PROGRAM TSTCON
C
C     VERSION 0.02     12 JULY 1984     L. SHOER
C
C     PROGRAM TO TEST CONSOLE INPUT AND OUTPUT
C
      BYTE INPUT,NEWCON,OLDCON,OUTPUT
C
50    WRITE(1,100)
100   FORMAT(' Console? ')
      READ(1,200)NEWCON
200   FORMAT(I2)
C
      CALL GETCON(OLDCON)
      WRITE(1,300)OLDCON,NEWCON
300   FORMAT(' OLDCON, NEWCON: ',2I10)
C
      WRITE(1,400)
400   FORMAT(' Input? ')
      READ(1,500)INPUT
500   FORMAT(I3)
C
      CALL SETCON(NEWCON)
      CALL CONOUT(INPUT)
      CALL SETCON(OLDCON)
      GO TO 50
C
      END
```

```
      PROGRAM RAMP
C
C     VERSION 0.09     12 FEBRUARY 1985     L. SHOER
C
C     PROGRAM TO INCREMENT THE VOLTAGE FROM A BSR DEVICE
C           VOLTAGE FIRST SET TO STARTING VAC
C           VOLTAGE INCREMENTED IN STEPS TO FINAL VOLTAGE
C
      BYTE ADCUNT,BELL,BRIGHT,DEVICE,DIM,JUNK,OFF,ON,TRMINL
C
      INTEGER ACTSTP,COUNTR,CYCLE1,DSRSTP,FNLVLT
      INTEGER MXSTPS,STEP,STPINC,STRVLT,TIME,TTLTIM
C
      REAL VLTINC
C
C     TERMINAL IN USE IS SPECIFIED BY TRMINL
           COMMON /TBYTE/TRMINL
           DATA TRMINL/05/
C
C     BSR FUNCTIONS
           DATA OFF, ON, DIM, BRIGHT/
     1           3,  2,   4,     5/
C
      DATA BELL/07/
C
C     NUMBER OF STEPS TO GO FROM 8 VAC TO 120 VAC
           MXSTPS=150
C
C     VOLTAGE INCREMENT FOR EACH STEP FROM 8 VAC TO 120 VAC
           VLTINC=112.0/FLOAT(MXSTPS)
C
C     CLEAR SCREEN
100        CALL CLRSCR
C
C     SIGN ON MESSAGE
           WRITE(1,200)
200        FORMAT(
     1          ' RAMP     version 0.09     L. Shoer',//
     2          ' This program will increment voltage output ',
     3          'on a designated control module',//
     4          '       -minimum starting voltage:  8 VAC',/
     5          '       -maximum final voltage:   120 VAC',/
     6          '       -actual starting and final voltages ',
     7          'are selectable between 8 and 120 VAC',///)
C
C     OBTAIN ADC-1 UNIT TO CONTROL
300        WRITE(1,400)
400        FORMAT(' ADC-1 unit to control? (1 or 2) [1] ')
           READ(1,500)ADCUNT
500        FORMAT(I5)
           IF (ADCUNT.EQ.0) ADCUNT=1
           IF ((ADCUNT.LT.1).OR.(ADCUNT.GT.2)) GO TO 300
C
C     OBTAIN DEVICE TO CONTROL
600        WRITE(1,700)
700        FORMAT(' Device to control? (1 to 32) [1]  ')
           READ(1,500)DEVICE
           IF (DEVICE.EQ.0) DEVICE=1
           IF ((DEVICE.LT.1).OR.(DEVICE.GT.32)) GO TO 600
C
C     OBTAIN STARTING VOLTAGE DESIRED
720        WRITE(1,730)
730        FORMAT(' Starting voltage desired? (8 to 120) [8] ')
           READ(1,500)STRVLT
```

```
              IF ((STRVLT.LT.8).OR.(STRVLT.GT.120)) GO TO 720
C
C       OBTAIN FINAL VOLTAGE DESIRED
800           WRITE(1,900)
900           FORMAT(' Final voltage desired? (8 to 120) [120] ')
              READ(1,500)FNLVLT
              IF (FNLVLT.EQ.0) FNLVLT=120
              IF ((FNLVLT.LT.8).OR.(FNLVLT.GT.120)) GO TO 800
C
C       ACTUAL STEPS REQUIRED TO GET TO FINAL VOLTAGE DESIRED
              ACTSTP=IFIX(FLOAT(FNLVLT-STRVLT)/VLTINC)
C       NUMBER OF STEPS DESIRED BETWEEN STRVLT AND FNLVLT
1000          WRITE(1,1100)STRVLT,FNLVLT,ACTSTP,ACTSTP
1100          FORMAT(
     1             ' Number of steps desired between',I4,' VAC and',
     2               I4,' VAC?',/
     3               ' Range = 1 to',I6,'  [',I6,']  ')
              READ(1,500)DSRSTP
              IF (DSRSTP.EQ.0) DSRSTP=ACTSTP
              IF ((DSRSTP.LT.1).OR.(DSRSTP.GT.ACTSTP)) GO TO 1000
C       NUMBER OF ACTUAL STEPS (BURST) IN EACH DESIRED STEP
              STPINC=ACTSTP/DSRSTP
C
C       TOTAL TIME FOR VOLTAGE RAMPING
1200          WRITE(1,1300)
1300          FORMAT(' Total time (minutes) for voltage ramping?  ')
              READ(1,500)TTLTIM
              IF (TTLTIM.LE.0) GO TO 1200
C       CONVERT TIME TO SECONDS
              TTLTIM=TTLTIM*60
C
C       WAIT BETWEEN EACH DESIRED STEP (WHICH INCLUDES STPINC ACTUAL STEPS)
C       STPINC ACTUAL STEPS IS CALLED A BURST
C       EACH ACTUAL STEP TAKES A MINIMUM OF 1.4 SECONDS FOR TRANSMISSION
C       ASSUME EACH ACTUAL STEP TAKES 3 SECONDS (SOME TIME FOR OVERHEAD, TOO)
              TIME = TTLTIM/DSRSTP - STPINC*3
              IF (TIME.LT.1) TIME=1
C
C
C       SETUP
C       CLEAR SCREEN
2000          CALL CLRSCR
C       POSITION CURSOR HALF-WAY DOWN SCREEN
              WRITE(1,2100)
2100          FORMAT(/////////////////)
C       TURN OFF BSR DEVICE
2200          CALL ACNTRL(ADCUNT,DEVICE,OFF,VALID)
              IF (.NOT.VALID) GO TO 4000
C       TURN ON BSR DEVICE
              CALL ACNTRL(ADCUNT,DEVICE,ON,VALID)
              IF (.NOT.VALID) GO TO 4000
C       DIM BSR DEVICE TO STARTING VOLTAGE
              DO 2900 CYCLE1=1,ACTSTP
C                    TELL USER WHAT IS HAPPENING
                          IF (MOD(CYCLE1,2).EQ.0) GO TO 2400
                          WRITE(1,2300)CYCLE1,ACTSTP
2300                      FORMAT('+',20X,'Calibration in progress',
     1                       5X,I4,'/',I4,18(' '))
                          GO TO 2600
2400                      WRITE(1,2500)
2500                      FORMAT('+',70(' '))
2600                CALL ACNTRL(ADCUNT,DEVICE,DIM,VALID)
                    IF (.NOT.VALID) GO TO 4000
2900          CONTINUE
C
*       BEGIN TO DO 1 LEVEL
```

```
3000            WRITE(1,3100)
3100            FORMAT('+',10X,'When ready to ramp voltage, ',
     1          'press any key',25(' '))
                CALL CONIN(JUNK)
C       RAMP VOLTAGE
C       GO THROUGH LOOP NUMBER OF STEPS DESIRED BY USER
C               SETUP
                        STEP=0
                        COUNTR=0
C               TELL USER WHICH STEP WE ARE ON
3200                    STEP=STEP+1
                        WRITE(1,3300)STEP,DSRSTP
3300                    FORMAT('+',30X,'Step',I5,' of',I5,30(' '))
C                       RAMP BURST
                        DO 3400 CYCLE1=1,STPINC
C                            INCREMENT COUNTER FOR NUMBER OF ACTUAL STEPS
                                COUNTR=COUNTR+1
C                            BRANCH OUT IF DONE
                                IF (COUNTR.GT.ACTSTP) GO TO 5000
C                            NOT DONE, INCREMENT
                                CALL ACNTRL(ADCUNT,DEVICE,BRIGHT,VALID)
                                IF (.NOT.VALID) GO TO 4000
3400                    CONTINUE
C                       WAIT BETWEEN EACH BURST OF INCREMENTING
                              CALL SLEEP(TIME)
                GO TO 3200
C
C       ACNTRL RETURNS FLAG SET NOT VALID
4000            WRITE(1,4100)BELL
4100            FORMAT(1X,A1,'ERROR: BSR control not possible',/)
C
5000    END
```

```
BC(PLOTAD);
IN(PLOTAD);
IN(BATESSYS);
SR(ADCLIB,A:PLIB1,A:AUXLIB,A:XFORLIB);
FI;
EN;
```

```
      PROGRAM PLOTAD
C
C     VERSION 0.07     17 MAY 1985     L. SHOER
C
C     PLOT ANALOG DATA FROM REMOTE MEASUREMENTS DEVICE
C
      BYTE ADCUNT,BELL,BL,BY,CHAN,CNTRLC,KEY,RANGE,SPCBAR
      LOGICAL VALID,INSTAM
      INTEGER DELAY,FRSPNT,LSTPNT,PNTR,RAWDAT
      REAL MINTIM,MAXTIM,MINVLT,MAXVLT
      REAL MLVOLT(3600),MNMLVL,MXMLVL,MVCON,TIME,TINC
C
C     PLOT VARIABLES
            BYTE XLBL(40),YLBL(40),TITLE(40)
C
C     LOAD TITLES
            DATA XLBL /3*'*','T','i','m','e',' ',
     1                '(','m','i','n',')',3*'*'/
            DATA YLBL /3*'*','M','i','l','l','i',
     1                'v','o','l','t','s',3*'*'/
            DATA TITLE /40*' '/
C
      DATA BELL, CNTRLC, SPCBAR/
     1       07,     03,     32/
C
      DATA BL, BY/
     1     'L','Y'/
C
C     ZERO DATA ARRAY
            DATA MLVOLT/3600*0.0/
C
C     LOAD MINIMUM AND MAXIMUM MILLIVOLT VALUES (WILL BE UPDATED)
            DATA  MNMLVL,   MXMLVL/
     1           +1.0E+30, -1.0E+30/
C
C     CLEAR SCREEN
100         CALL CLRSCR
C
C     SIGN ON MESSAGE
            WRITE(1,200)
200         FORMAT(
     1            ' PLOTAD     version 0.07     L. Shoer',//
     2            ' This program will plot voltage vs. ',
     3            'time from a designated ADC-1 analog channel',//
     4            ' Voltage range is:',/
     5            '       L)ow:    -0.04096 to +0.04096 volts',/
     7            '       H)igh:   -0.4096  to +0.4096 volts',//)
C
C     OBTAIN ADC-1 UNIT TO CONTROL
300         WRITE(1,400)
400         FORMAT(' ADC-1 unit to control? (1 or 2) [1] ')
            READ(1,500) ADCUNT
500         FORMAT(I2)
            IF (ADCUNT.EQ.0) ADCUNT=1
            IF ((ADCUNT.LT.1).OR.(ADCUNT.GT.2)) GO TO 300
C
C     OBTAIN ADC-1 ANALOG CHANNEL
600         WRITE(1,700)
700         FORMAT(
     1          ' Analog channel to monitor?',/
     2          '      L)ow range:  available channels are 1 to 8',/
     3          '      H)igh range: available channels are 1 to 16',/
     4          ' Channel? [1]  ')
```

```
              READ(1,500) CHAN
              IF (CHAN.EQ.0) CHAN=1
              IF ((CHAN.LT.1).OR.(CHAN.GT.16)) GO TO 600
C
C       OBTAIN VOLTAGE RANGE
C             SET DEFAULT, HIGH RANGE
800                 INSTAM = .FALSE.
C             PROMPT USER FOR RANGE
900                 WRITE(1,1000)
1000                FORMAT(' Voltage range (H or L)? [H]  ')
                    READ(1,1100) RANGE
1100                FORMAT(1A1)
                    IF (RANGE.EQ.BL) INSTAM=.TRUE.
                    IF (INSTAM.AND.(CHAN.GT.8)) GO TO 600
C
C       OBTAIN TIME BETWEEN DATA POINT COLLECTION
2000        WRITE(1,2100)
2100        FORMAT(' Time betwen each data point collected (SEC)?  ')
            READ(1,2150) DELAY
2150        FORMAT(I5)
            IF (DELAY.LE.0) GO TO 2000
C
C       OBTAIN TOTAL TIME FOR DATA COLLECTION
2200          WRITE(1,2300)
2300          FORMAT(' Collect data for how many MINUTES?  ')
              READ(1,2400) MAXTIM
2400          FORMAT(F10.0)
              IF (MAXTIM.LE.0.0) GO TO 2200
C
C       SET UP PARAMETERS
C       MILLIVOLT AXIS
C             DEFAULT TO HIGH RANGE
                    MINVLT = -410.0
                    MAXVLT = +410.0
                    MVCON = 0.10
C             LOW RANGE IF INSTRUMENTATION AMPLFIER IS ON
                    IF (.NOT.INSTAM) GO TO 3000
                    MINVLT = -41.0
                    MAXVLT = +41.0
                    MVCON= 0.010
C       TIME AXIS
C             BEGINNING TIME OF PLOT
3000                MINTIM = 0.0
C             ENDING TIME OF PLOT
C                   MAXTIM = USER ENTRY
C             FIRST POINT TO BE PLOTTED
                    FRSPNT = 1
C             LAST POINT TO BE PLOTTED
                    LSTPNT = IFIX(MAXTIM)*60 / DELAY + 1
C             INNCREMENT BETWEEN EACH TIME VALUE (MINUTES)
                    TINC = FLOAT(DELAY)/60.
C             MAKE CERTAIN THERE IS ENOUGH ROOM FOR ALL THE DATA
                    IF (LSTPNT.GT.3600) GO TO 2000
C
C       SET UP PLOTTER
              CALL PINCR (0,0,0,1.0,RAWDAT,0,
     1                    MINTIM,MAXTIM,MINVLT,MAXVLT,
     2                    XLBL,YLBL,TITLE)
C
C       TRACK ANALOG VALUE UNTIL USER INDICATES PLOTTING TO START
C             TELL USER WHAT TO DO
4000                CALL HOME
                    WRITE(1,4100)
4100                FORMAT('+Press any key when plotting should ',
     1                     'start',20(' '),///)
C             TRACK ANALOG VALUE
```

```
C                   INITIAL TIME VALUE
                        TIME = 0.0
C                   GET A MILLIVOLT VALUE
4200                    CALL ADCONV(ADCUNT,CHAN,INSTAM,DATUM,VALID)
                        IF (VALID) GO TO 4500
C                       ERROR CONDITION - DATUM IS NOT VALID
                            CALL HOME
                            WRITE(1,4300) BELL
4300                        FORMAT('+',1A1,'ERROR: ',
     1                      'Analog-to-digital converter ',
     2                      'indicates an error')
                            GO TO 4600
C                   SCALE VOLTAGE TO MILLIVOLTS
4500                    MLVOLT(1) = FLOAT(DATUM) * MVCON
C                   PLOT POINT (PEN UP)
                        CALL PPOINT (0,TIME,MLVOLT(1))
C                   IF PLOTTING SHOULD START, JUMP OUT
4600                    CALL CONIN1 (KEY)
                        IF (KEY.EQ.CNTRLC) GO TO 7000
                        IF (KEY.NE.0) GO TO 5000
C                   PLOTTING SHOULD NOT START YET
C                   DELAY BEFORE GETTING ANOTHER POINT
                        CALL SLEEP (DELAY)
C                   TRACK ANALOG VALUE
                        GO TO 4200
C
C       START PLOTTING
C               INDICATE HOW TO STOP DATA COLLECTION EARLY
5000                    CALL HOME
                        CALL CLRALP
                        WRITE(1,5100)
5100                    FORMAT(
     1                      '+To stop data collection early, ',
     2                      ' press space bar.',19(' '),///)
C               INITIALIZE
                        TIME = -TINC
                        CALL PPOINT (0,0.0,MLVOLT(1))
        DO 5500 PNTR=FRSPNT,LSTPNT
C               CALCULATE TIME VALUE
                        TIME=TIME+TINC
C               SEE IF EARLY STOP TO DATA COLLECTION IS REQUESTED
5300                    CALL CONIN1 (KEY)
                        IF((KEY.EQ.SPCBAR).OR.(KEY.EQ.CNTRLC))GO TO 5700
C               GET A VOLTAGE
                        CALL ADCONV (ADCUNT,CHAN,INSTAM,DATUM,VALID)
                        IF (VALID) GO TO 5400
C                       ERROR CONDITION
                            CALL HOME
                            WRITE(1,4300) BELL
                            I = PNTR - 1
                            CALL PPOINT (0,TIME,MLVOLT(I))
                            GO TO 5300
C               SCALE VOLTAGE VALUE
5400                    MLVOLT(PNTR) = FLOAT(DATUM) * MVCON
C               UPDATE MINIMUM AND MAXIMUM MILLIVOLT VALUES
                    IF (MLVOLT(PNTR).LT.MNMLVL) MNMLVL=MLVOLT(PNTR)
                    IF (MLVOLT(PNTR).GT.MXMLVL) MXMLVL=MLVOLT(PNTR)
C               PLOT POINT
                        CALL PPOINT (1,TIME,MLVOLT(PNTR))
C               DELAY BEFORE GETTING ANOTHER POINT
                        CALL SLEEP (DELAY)
5500    CONTINUE
C
C       SEE IF FURTHER PLOTTING OF THIS DATA IS DESIRED
5700            CALL HOME
                CALL CLRALP
```

```
              WRITE(1,5800)
5800          FORMAT(
     1               '+',70(' '),/,
     2               '+Replot the same data (Y or N)?  [N]  ')
              READ(1,1100) KEY
              IF (KEY.NE.BY) GO TO 7000
C       FURTHER PLOTTING IS DESIRED
C             IF DATA COLLECTION ENDED EARLY, UPDATE VALUES
              IF (PNTR.GE.LSTPNT) GO TO 5900
                     LSTPNT = PNTR - 1
                     MAXTIM = FLOAT((LSTPNT-1) * DELAY) / 60.0
C             IF PLOT LIMITS WILL NOT WORK, END PROGRAM
                     IF (FRSPNT.GE.LSTPNT) GO TO 7000
                     IF (MNMLVL.GE.MXMLVL) GO TO 7000
C             PLOT DATA
C             SET UP PLOTTER
5900                 CALL PINCR (0,0,0,1.0,RAWDAT,0,
     1                           MINTIM,MAXTIM,MNMLVL,MXMLVL,
     2                           XLBL,YLBL,TITLE)
C             PLOT POINTS
                     TIME = -TINC
                     CALL PPOINT (0,0.0,MLVOLT(1))
                     DO 6000 PNTR = FRSPNT, LSTPNT
                            TIME = TIME + TINC
                            CALL PPOINT (1,TIME,MLVOLT(PNTR))
6000                 CONTINUE
       GO TO 5700
C
C      CLEAR SCREEN
7000          CALL CLRSCR
C
       END
```

```
      PROGRAM EXP1
C
C     RMS EXPERIMENT NUMBER 1
C
C     B.BURLEIGH      11-84
C
C     The object of this experiment is to use the RMS device
C     to monitor light intensity. The console speaker is to be
C     driven at a rate proportional to light intensity.
C
      BYTE    ADCUNT,CHAN,BELL
      INTEGER DATUM,IY
      LOGICAL VALID,INSTAM
C
C     PLOT VARIABLES
            BYTE KEY,TRMINL
            BYTE XLBL(40),YLBL(40),TITLE(40)
C
      INTEGER IRES,OPTION
C
      REAL XSTART,XFIN,XINC
      REAL AMIN,AMAX
      REAL BMIN,BMAX
      REAL Y,X
C
      COMMON  /TBYTE/TRMINL
C
C     LOAD TITLES
            DATA XLBL/3*'*','X','L','A','B','E','L',3*'*'/
            DATA YLBL/3*'*','Y','L','A','B','E','L',3*'*'/
            DATA TITLE/'T','I','T','L','E'/
C
      DATA BELL /07/
C
C     TERMINAL
C           OPTIONS:
C                 2 = LEAR SIEGLER ADM5
C                 4 = QUME
C                 5 = VISUAL 500
C           ASSIGNMENT
                  TRMINL = 2
C
C     SET UP PLOT PARAMETERS
            XSTRT=1.00
            XFIN=100.0
            XINC=1.0
C
            AMIN=0.0
            AMAX=100.0
C
            BMIN=0.0
            BMAX=4100.0
C
C     WE WILL USE ADC UNIT #1
      ADCUNT = 1
C
C     WE WILL USE CHANNEL NUMBER 1
      CHAN = 1
C
C     NO INSTRUMENT AMP
      INSTAM = .FALSE.
C
C     SET UP PLOTTER
            CALL PINCR(0,0,0,1.0,IY,0,
```

```
C
C       PUT PEN AT 0,0 AND INIT X AND Y
                CALL PPOINT(0,0.0,0.0)
C
                X=0.0
C
C       GET A VALUE FOR THE POTENTIAL
5               CALL ADCONV(ADCUNT,CHAN,INSTAM,DATUM,VALID)
C
                IF(.NOT.VALID) GO TO 100
C
C       ASSIGN DATUM TO Y AND INCRIMENT TIME (X)
                X=X+1.0
                Y=FLOAT(DATUM)
        WRITE(2,10)X,Y
10      FORMAT(' ',F10.0,F10.0)
C
C       PLOT POINT
                CALL PPOINT(1,X,Y)
C
        CALL SLEEP(1)
        GO TO 5
C
C       ERROR CONDITION
100             WRITE(1,110)
110             FORMAT('ERROR - OVERFLOW?? ')
C
        GO TO 5
        END
```

```
      PROGRAM EXP1
C
C     RMS EXPERIMENT NUMBER 1
C
C     B.BURLEIGH      11-84
C
C     The object of this experiment is to use the RMS device
C     to monitor light intensity. The console speaker is to be
C     driven at a rate proportional to light intensity.
C
      BYTE    ADCUNT,CHAN,BELL
      INTEGER DATUM
      LOGICAL VALID,INSTAM
C
      DATA BELL /07/
C
C     WE WILL USE ADC UNIT #1
      ADCUNT = 1
C
C     WE WILL USE CHANNEL NUMBER 1
      CHAN = 1
C
C     NO INSTRUMENT AMP
      INSTAM = .FALSE.
C
C     THE VALUE OF THE RESISTANCE ACROSS THE PHOTO-SENSITIVE
C     CELL WILL VARY WITH THE LIGHT INTENSITY. IF WE APPLY A
C     POTENTIAL ACROSS IT WITH A RESISTOR IN SERIES WE CAN MEASURE
C     A VOLTAGE ACROSS ITS LEGS INVERSELY PROPORTIOONAL TO INTENSITY.
C
C     GET A VALUE FOR THE POTENTIAL
5           CALL ADCONV(ADCUNT,CHAN,INSTAM,DATUM,VALID)
C
            IF(.NOT.VALID) GO TO 100
C
C     KILL SOME AMOUNT OF TIME PROPORTIONAL TO DATUM
            DO 10 I=1,50
                  DO 9 J=1,DATUM
9                 CONTINUE
10          CONTINUE
C
            WRITE(1,20)BELL
20          FORMAT(1X,A1)
C
      GO TO 5
C
C     ERROR CONDITION
100         WRITE(1,110)
110         FORMAT('ERROR - OVERFLOW?? ')
C
      GO TO 5
      END
```

```
      PROGRAM EXP1
C
C     RMS EXPERIMENT NUMBER 1
C
C     B.BURLEIGH      11-84
C
C     The object of this experiment is to use the RMS device
C     to monitor light intensity. The console speaker is to be
C     driven at a rate proportional to light intensity.
C
      BYTE    ADCUNT,CHAN,BELL
      INTEGER DATUM,IY
      LOGICAL VALID,INSTAM
C
C     PLOT VARIABLES
            BYTE KEY,TRMINL
            BYTE XLBL(40),YLBL(40),TITLE(40)
C
      INTEGER IRES,OPTION
C
      REAL XSTART,XFIN,XINC
      REAL AMIN,AMAX
      REAL BMIN,BMAX
      REAL Y,X
C
      COMMON  /TBYTE/TRMINL
C
C     LOAD TITLES
            DATA XLBL/3*'*','X','L','A','B','E','L',3*'*'/
            DATA YLBL/3*'*','Y','L','A','B','E','L',3*'*'/
            DATA TITLE/'T','I','T','L','E'/
C
      DATA BELL /07/
C
C     TERMINAL
C           OPTIONS:
C                 2 = LEAR SIEGLER ADM5
C                 4 = QUME
C                 5 = VISUAL 500
C           ASSIGNMENT
                  TRMINL = 2
C
C     SET UP PLOT PARAMETERS
            XSTRT=1.00
            XFIN=100.0
            XINC=1.0
C
            AMIN=0.0
            AMAX=100.0
C
            BMIN=0.0
            BMAX=4100.0
C
C     WE WILL USE ADC UNIT #1
      ADCUNT = 1
C
C     WE WILL USE CHANNEL NUMBER 1
      CHAN = 1
C
C     NO INSTRUMENT AMP
      INSTAM = .FALSE.
C
C     SET UP PLOTTER
            CALL PINCR(0,0,0,1.0,IY,0,
```

```
C
C       PUT PEN AT 0,0 AND INIT X AND Y
              CALL PPOINT(0,0.0,0.0)
C
              X=0.0
C
C       GET A VALUE FOR THE POTENTIAL
5             CALL ADCONV(ADCUNT,CHAN,INSTAM,DATUM,VALID)
C
              IF(.NOT.VALID) GO TO 100
C
C       ASSIGN DATUM TO Y AND INCRIMENT TIME (X)
              X=X+1.0
              Y=FLOAT(DATUM)
        WRITE(2,10)X,Y
10      FORMAT(' ',F10.0,F10.0)
C
C       PLOT POINT
              CALL PPOINT(1,X,Y)
C
        CALL SLEEP(1)
        GO TO 5
C
C       ERROR CONDITION
100           WRITE(1,110)
110           FORMAT('ERROR - OVERFLOW?? ')
C
        GO TO 5
        END
```

```
      PROGRAM EXP2
C
C     RMS EXPERIMENT NUMBER 2
C
C     B.BURLEIGH      12-84
C
C     The object of this experiment is to use the RMS device
C     to monitor the resistance of a thermistor and drive a
C     heating device (light) to reach some pre-decided temp.
C
      BYTE    ADCUNT,CHAN,BELL,DEVICE,FNCTN
      INTEGER DATUM,TARGET
      LOGICAL VALID,INSTAM
C
C     WE WILL USE ADC UNIT #1
      ADCUNT = 1
C
C     WE WILL USE CHANNEL NUMBER 1 TO READ THERMISTOR
      CHAN = 10
C
C     NO INSTRUMENT AMP
      INSTAM = .FALSE.
C
C     SET UP PLOTTER
C     USE BSR UNIT ONE FOR THE LIGHT
      DEVICE = 2
C
C     GET A TARGET DATUM
            WRITE(1,1)
1           FORMAT(' WHAT IS THE TARGET (0-4095) ')
C
            READ(1,2)TARGET
2           FORMAT(I4)
C
C     GET A VALUE FOR THE POTENTIAL
5           CALL ADCONV(ADCUNT,CHAN,INSTAM,DATUM,VALID)
C
            IF(.NOT.VALID) GO TO 100
C
C     VOLTAGE ACROSS THE THERMISTOR IS HIGH WHEN THE RESISTANCE
C     IS LOW. THIS CORRESPONDS TO A COLD THERMISTOR!
            IF (DATUM.GT.TARGET) GO TO 20
C
C           DATUM LOW >>> DIM THE LIGHT TO RAISE THE POTENTIAL
C     ***
      WRITE(1,200)DATUM
200   FORMAT(' DIM! ',I4)
C     ****
                  FNCTN = 4
                  CALL ACNTRL(ADCUNT,DEVICE,FNCTN,VALID)
                  IF (.NOT.VALID) GO TO 120
                  GO TO 5
C
C           DATUM HIGH >>> BRIGHTEN LIGHT
20                FNCTN = 5
C     ****
      WRITE(1,210)DATUM
210   FORMAT(' BRIGHT ',I4)
C     ****
                  CALL   ACNTRL(ADCUNT,DEVICE,FNCTN,VALID)
                  IF(.NOT.VALID) GO TO 120
                  GO TO 5
C
C
```

```
100             WRITE(1,110)
110             FORMAT('ERROR -INPUT- OVERFLOW?? ')
                GO TO 5
C
120             WRITE(1,130)
130             FORMAT(' ERROR -WRITING')
                GO TO 5
        END
```